Classification

Agenda

- Problem definition
- Generative vs discriminative approaches
- Evaluation metrics
- Representation
- Feature engineering

Problem definition

Classification

- Find an appropriate decision function $f(\mathbf{x})$ to predict one or more categorical response variables $\mathbf{y}$
    - binary
    - muticlass
    - multilabel

Misclassification rates

- Let

$$I\left(Y, f(\mathbf{x})\right) = \begin{cases} 0, & \text{if } y = f(\mathbf{x}) \\ 1, & \text{otherwise} \end{cases}$$

分類正確

分錯

- The misclassification rate can be defined as

$$P\left(I\left(Y, f(\mathbf{x})\right) = 1\right)$$

分錯的機率

$$\simeq \frac{1}{n}\sum_{i=1}^{n} I(y_i, f(x_i))$$

Misclassification rates

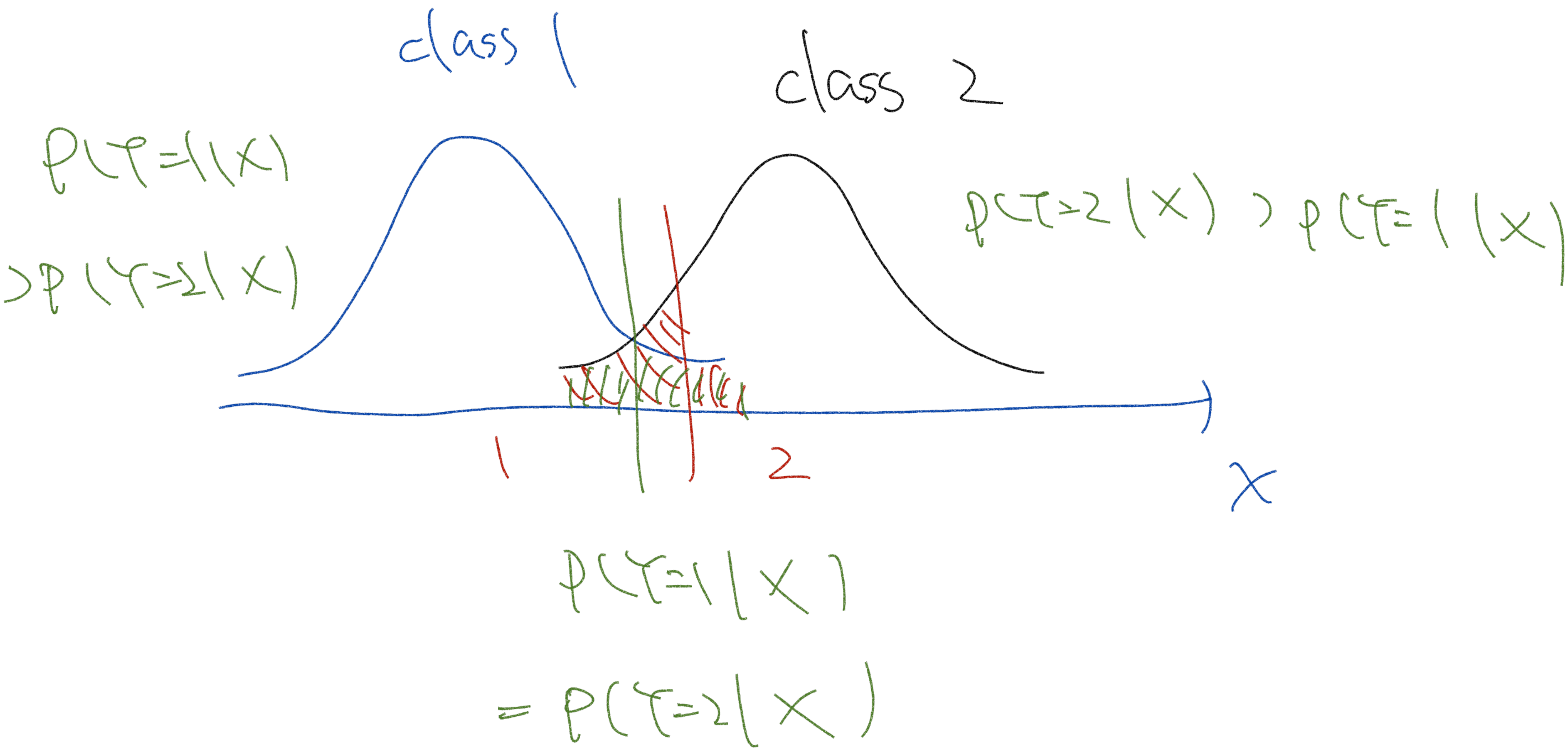
- Intuitively, one may learn an decision function $f(\mathbf{x})$ that minimizes misclassification rate.

$$\min_f \frac{1}{n}\sum_{i=1}^{n} I(y_i, f(x_i)) + \|f\|$$

- Unfortunately, loss function composited by step functions are difficult to optimize.

$$\frac{\partial I}{\partial f} = 0 \quad \text{forever}$$

Probabilistic interpretation



Generative vs discriminative

Generative models

- From Bayes theorem we have

$$P(Y \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid Y) \times P(Y)}{P(\mathbf{X})}$$
$$\propto P(\mathbf{X} \mid Y) \times P(Y)$$

- The decision function can be specified as

$$f(\mathbf{X}) = \arg \max_{Y} P(Y \mid \mathbf{X})$$
$$= \arg \max_{Y} P(\mathbf{X} \mid Y) \times P(Y)$$

Generative models

- The prior probability $P(Y)$ can be assigned in priori or be estimated empirically by n_Y/N

- Probabilistic model assumptions on $P(\mathbf{X} | Y)$:

  - $N\left(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}\right)$: linear discriminant analysis (LDA)

  - $N\left(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y\right)$: quadratic discriminant analysis (QDA)

  - Naive bayes

  - Nonparametric estimation: k–nearest neighbors

Discriminative models

- Learns the <u>decision boundaries</u> directly
- Less model assumption
- Loss function formulation
- More preferable in machine learning society

Logistic regression

- Softmax function:

$$\sigma(\mathbf{x}) = \frac{\exp\left[\beta_0 + \mathbf{x}^\top \boldsymbol{\beta}\right]}{1 + \exp\left[\beta_0 + \mathbf{x}^\top \boldsymbol{\beta}\right]}$$

- Decision function:

$$f(\mathbf{x}) = I\left(\sigma(\mathbf{x}) > \frac{1}{2}\right)$$

- Decision boundary: $\left\{\mathbf{x} : \sigma(\mathbf{x}) = \frac{1}{2}\right\}$

Multiclass logistic regression

- Assume that $y \in \{1, 2, \cdots, c\}$

- In multiclass logistic regression, we assume that

$$Y \overset{iid}{\sim} \text{Categorical}\left(\sigma_j(\mathbf{x})\right)$$

with softmax function

$$\sigma_k(\mathbf{x}) = \frac{\exp\left[\beta_{0k} + \mathbf{x}^\top \boldsymbol{\beta}_k\right]}{\sum_{j=1}^{c} \exp\left[\beta_{0k} + \mathbf{x}^\top \boldsymbol{\beta}_j\right]} \quad \overset{?}{=} P(Y=k|X)$$

representation

Multiclass logistic regression

- The likelihood function becomes

$$L = \prod_{i=1}^{N} \left[\prod_{k=1}^{c} \sigma_k(\mathbf{x}_i) \right]^{y_i^{(k)}}$$

只有1個是1

其他都是0

where

$$y_i^{(k)} = \begin{cases} 1, & \text{if } y_i = k \\ 0, & \text{otherwise} \end{cases}$$

, $k = 1, \ldots, c$

c 個 $y_i^{(k)}$ for $i = 1, \ldots, N$

Multiclass logistic regression

- The log-likelihood becomes

$$\ell = \sum_{i=1}^{N} \left[\sum_{k=1}^{c} y_i^{(k)} \log \sigma_k(\mathbf{x}_i) \right]$$

- $-\sum_{k=1}^{c} y_i^{(k)} \log \sigma_k(\mathbf{x})$ is also called cross entropy
- What will happen if c=2? (homework)

$f(x) = \arg\max_{k} \sigma_k(x)$

Metrics

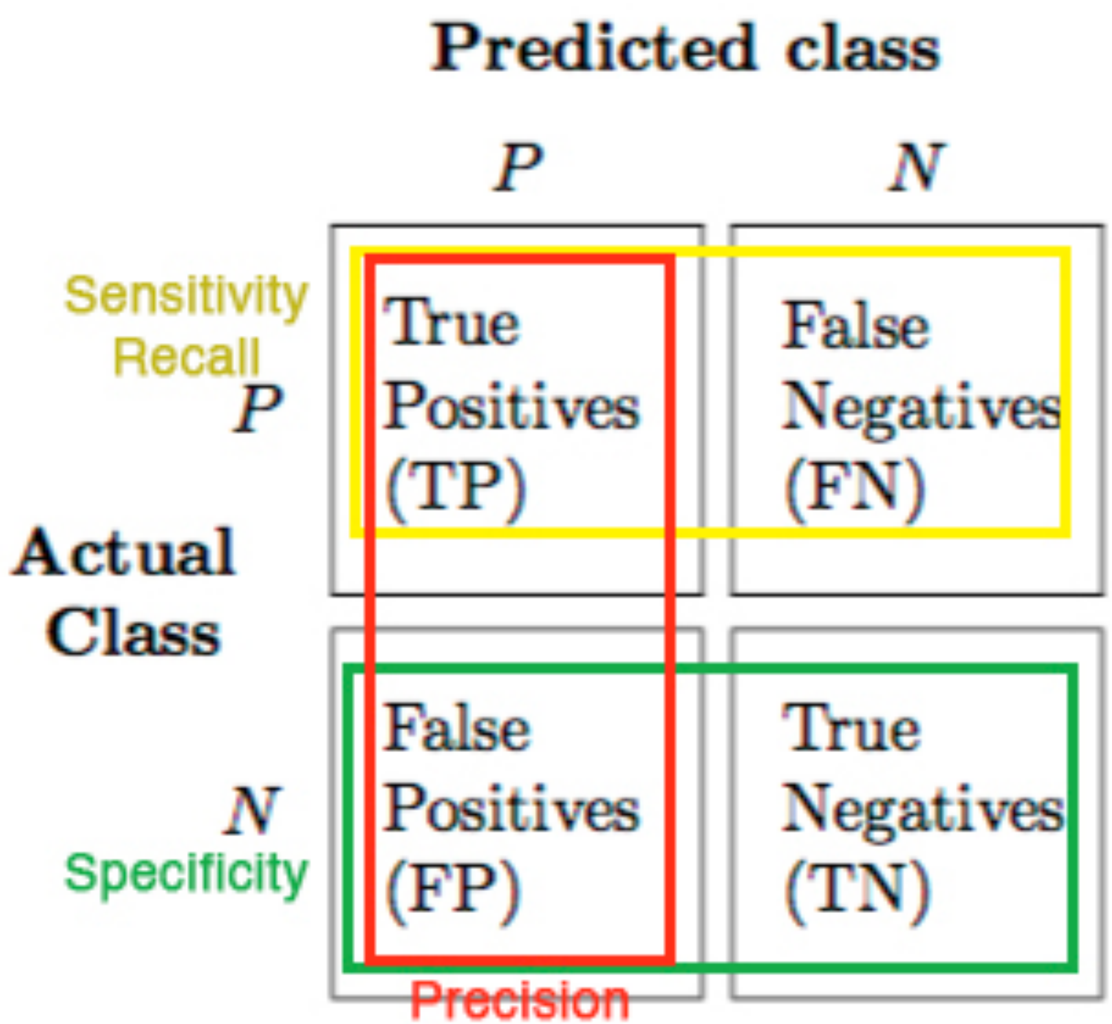
Binary classification

- Confusion matrix

True positives (TP)	False Negatives (FN)
False Positives (FP)	True Negatives (TN)

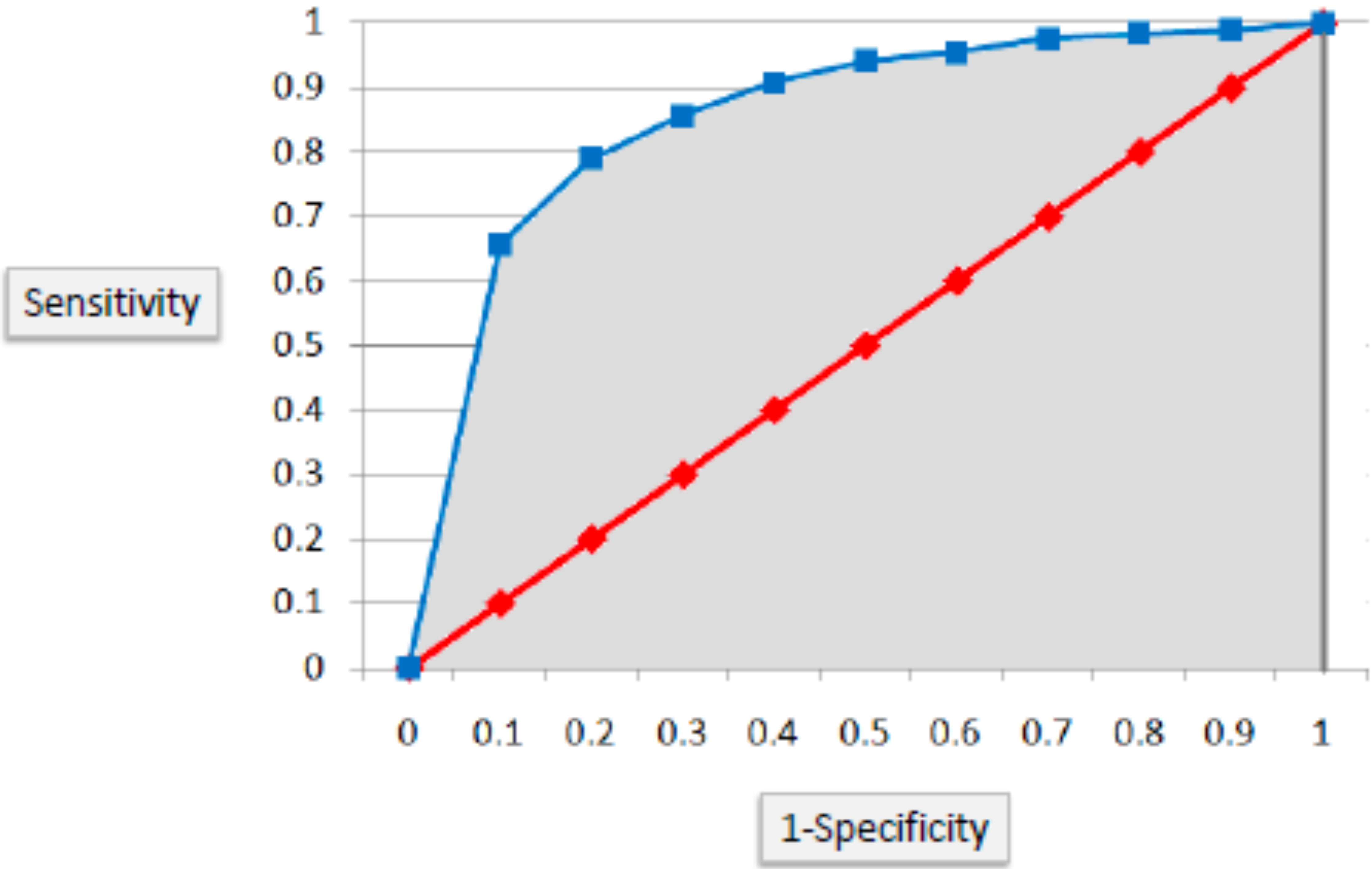
- Misclassification rate = (FN + FP) / N
- Sensitivity (recall) = TP / (TP+FN)
- Precision = TP / (TP + FP)

- Specificity = TN / (TN+FP)

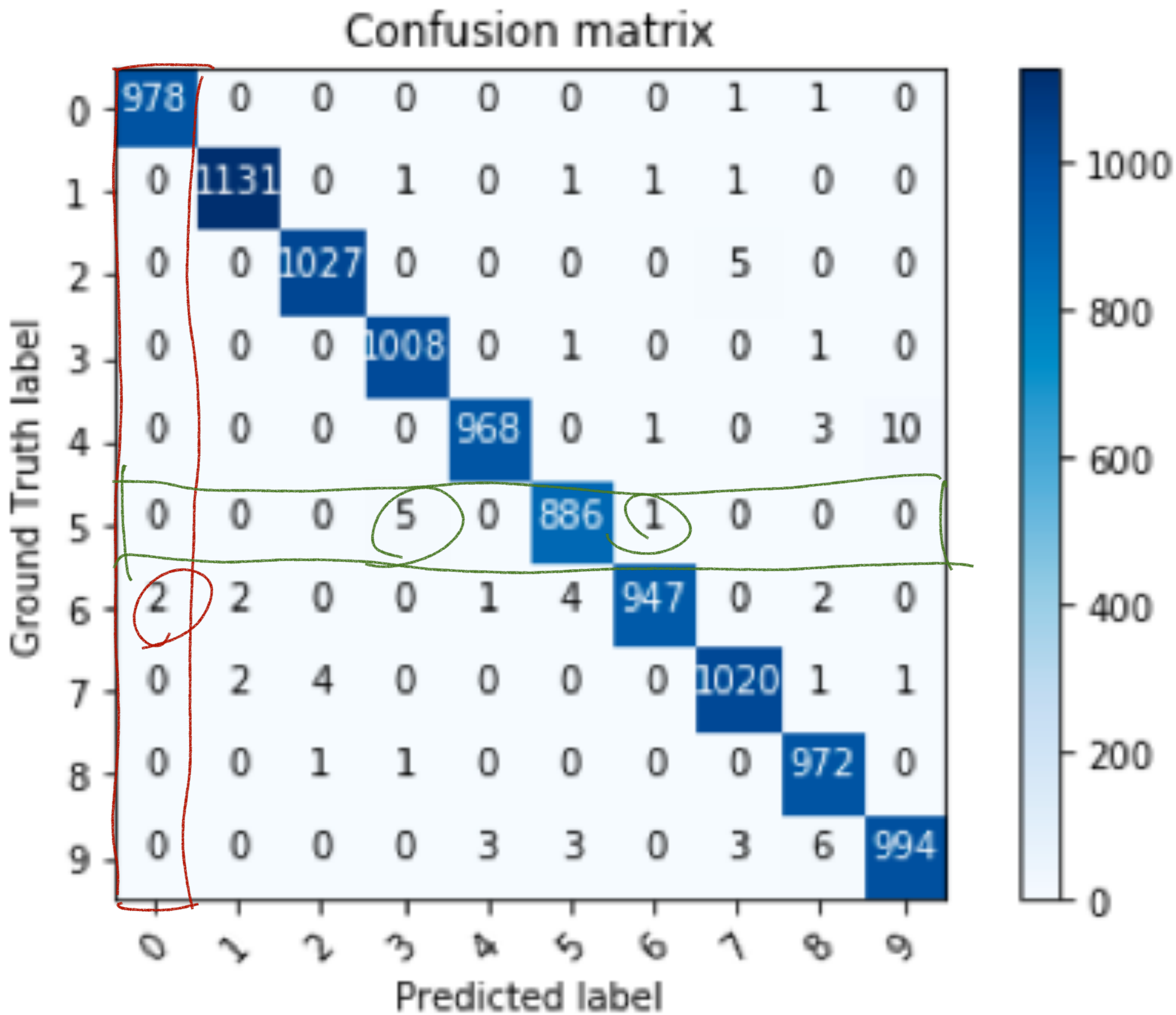


- F1-score is a single metric that combines both precision and recall via their harmonic mean

Area under ROC



Multiclass classification



- precision, recall and F1-score
  - micro: calculate metrics globally by counting the total number of times each class was correctly predicted and incorrectly predicted
  - macro: calculate metrics for each "class" independently, and find their unweighted mean. This does not take label imbalance into account.

Representations

- Linear function $\beta_0 + \mathbf{x}^\top \boldsymbol{\beta}_k$
- Trees (e.g. random forest, gradient boosting trees, etc)
- Kernel tricks
- Deep neural networks
- etc.

Feature engineering

- Sometimes we may need to transform the raw data (e.g. images) to some useful features (e.g. breast cancer dataset)
- In the past decades the transformations are carried out by human intelligent
- On of the most appealing advantage of deep learning (especially CNN) is that such transformations can be determined automatically

Readings

- Chapter 17 of "Computational and Inferential Thinking"

- Classification Metrics

- Chapter 4 of "Machine Learning with TensorFlow"

- Tensorflow playground

Homework

1. Implement multiclass logistic regression on your own by (stochastic) gradient descent and apply it to the wine dataset

2. Compare your result with that obtained by sklearn.linear_model.LogisticRegression with a very large C and multi_class='multinomial'

3. Evaluate the classification performances by micro/macro precisions, recalls, and F1 scores
(by cross-validation)